Efficient Bulk Data Replication for the Earth System Grid

Alex Sim[1], Dan Gunter[1], Vijaya Natarajan[1], Arie Shoshani[1], Dean Williams[2], Jeff Long[2], Jason Hick[3], Jason Lee[3], Eli Dart[4]

[1] Lawrence Berkeley National Laboratory
[2] Lawrence Livermore National Laboratory
[3] National Energy Research Scientific Computing Center
[4] Energy Sciences Network

Abstract

The Earth System Grid (ESG) community faces the difficult challenge of managing the distribution of massive data sets to thousands of scientists around the world. To move data replicas efficiently, the ESG has developed a data transfer management tool called the Bulk Data Mover (BDM). We describe the performance results of the current system and plans towards extending the techniques developed so far for the upcoming project, in which the ESG will employ advanced networks to move multi-TB datasets with the ultimate goal of helping researchers understand climate change and its potential impacts on world ecology and society.

Keywords
ESG, climate, BDM, data transfer, NetLogger

1. Introduction

The Earth System Grid (ESG) [1] community faces the difficult challenge of managing the distribution of massive data sets to thousands of scientists around the world. An important new collection of climate data sets, referred to as the "centralized data", is expected to comprise 0.6-1.2PB during the Intergovernmental Panel on Climate Change (IPCC) Fifth Assessment Report (AR5) in 2011. The new centralized data can only be efficiently served to researchers all over the world by replicating it to sites closer to them. To move data replicas efficiently, the ESG has developed a data transfer management tool called the Bulk Data Mover (BDM) [2]. The BDM achieves high performance using a variety of techniques, including multi-threaded concurrent transfer connections, data channel caching, balanced transfer server connections, storage I/O prefetching, and automatic transfer tuning. To optimize the BDM to use the full available

full available network and storage bandwidth of multi-gigabit paths, monitoring information from the BDM and transfer servers is collected and analyzed in near-real time with the NetLogger toolkit. In the results from ESG data replication on the current networks, NetLogger [4] analyses provided an intuitive view of time-varying patterns in the overlap between multiple concurrent transfers that proved invaluable for tuning the BDM concurrency algorithms. In future networks, it is expected that tuning and debugging concurrent performance will become even more difficult. As data volume grows super-linearly and network and storage bandwidth increase, more concurrency will be needed for data transfers, the volume of monitoring data will increase, and disparities between components will be exacerbated. We describe the performance results of the current system and plans towards extending the techniques developed so far for the upcoming project, in which the ESG will employ advanced networks to move multi-TB datasets with the ultimate goal of helping researchers understand climate change and its potential impacts on world ecology and society.

2. Earth System Grid

As the climate community makes its first steps towards building a "science gateway"—a data access and analysis system open to everyone —the *"Earth System Grid" (ESG)* is central to the current and future infrastructure that enables the large federated enterprise system for the dissemination and management of extreme scale climate resources. ESG provides climate resources such as data, information, models, analysis and visualization tools, and other computational capabilities for data management and diagnosis. The ESG project's goals are (1) to make data more useful to climate researchers by developing Grid technology that enhances data usability; (2) to meet specific needs which national and international climate projects have for distributed database, data access, and data movement; (3) to provide a universal and secure web-based data access portal for broad-based multi-model data collections; and (4) to provide a wide-range of Grid-enabled climate data analysis tools and diagnostic methods to climate communities [3]. Thus, ESG is working to integrate distributed data and computers, high-bandwidth wide-area networks, and remote computing using climate data analysis tools in a highly collaborative problem-solving environment.

Since production began in 2004, the ESG has hosted and distributed significant and often very large data collections for many well-known efforts in climate science. As of February 2010, the ESG production system has over 16,000 registered users. ESG manages approximately 270 TB of

model data, comprising the contents of archives at five sites around the US. ESG users have downloaded more than 1PB of data.

3. Bulk Data Mover

The Bulk Data Mover (BDM) is responsible for the successful replication of large datasets. Climate datasets are characterized by large numbers of small files; to handle this issue the ESG uses the BDM software as a higher-level data transfer management component to manage the file transfers with optimized transfer queue and concurrency management algorithms.

The BDM can accept a request composed of multiple files or an entire directory. The files or directory are described as Universal Resource Locators (URLs) that indicate the source sites that contain the files. The request also contains the target site and directory where the replicated files will reside. If a directory is provided at the source, then the BDM will replicate the structure of the source directory at the target site. The BDM is capable of transferring multiples files concurrently as well as using parallel TCP streams. The optimal level of concurrency or parallel streams is dependent on the bandwidth capacity of the storage systems at both ends of the transfer as well as achievable bandwidth on the wide-area-network (WAN). Setting up the level of concurrency correctly is an important issue, especially in climate datasets, because of the small files. Concurrency that is too high becomes ineffective (high overheads and increased congestion), and concurrency that is too low will not take advantage of available bandwidth. A similar phenomenon was observed when setting up the level of parallel streams.

The BDM is designed to work in a “pull mode”, where the BDM runs as a client at the target site. This choice is made because of practical security aspects: site managers usually prefer to be in charge of pulling data, rather than having data pushed at them. However, the BDM could also be designed to operate in a “push mode”, or as an independent third-party service. Because a large scale data replication can take a long time (from many minutes to hours and even days) the BDM must be an asynchronous service. That means that when a replication request is launched, a “request token” is returned to the client. The client should be able to use that request token to check the status of the request execution at any time. Another obvious implication to the long lasting nature of large scale replication is the need for automatic monitoring and recovery from any transient failures, which is an important part of the BDM's design.

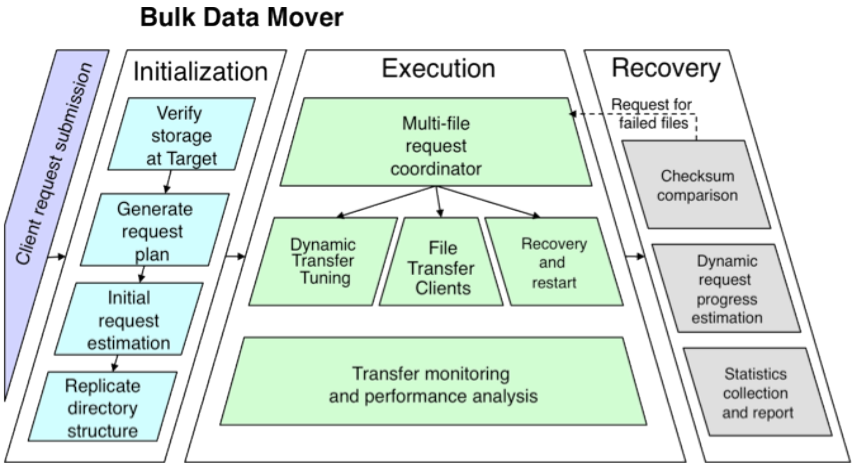

Figure 1. The design of the Bulk Data Mover (BDM)

3.1 Multi-phase Transfer Request Management

The tasks that the BDM performs to accomplish a successful replication are organized into three phases, as shown in Figure 1. The initialization phase plans and prepares file replications from the data source to the local target storage. It includes the following tasks: 1) Storage allocation verification at the target site; this requires collecting the total data size from the source site. 2) Generating a request plan. The plan includes the initial level of concurrency, number of parallel streams, and buffer size for the request. 3) Returning an initial request estimation to the client. 4) Mirroring the directory structure of the source at the target site. It then generates an execution plan that includes pair-wise source-to-target URLs for all the files to be replicated. This is used by the execution phase.

The Execution phase transfers the requested files, while monitoring and analyzing transfer performance for dynamic adjustment on the transfer properties. It consists of four modules. 1) The Multi-File Request Coordinator uses the information from the "execution plan" and transfer properties including the concurrency level, and accordingly instantiates the file transfer client. 2) The File Transfer Client can support any transfer protocols or services preferred by the virtual organization. Supported transfer protocol could be GridFTP, HTTPS, SCP, SFTP, etc. 3) The Recovery and Restart module continuously monitors the health of the system and the files being transferred. If a transient error occurs it waits for the system to recover and depending on the transfer protocol used, either removes the partial files transferred and reschedules the transfer or continues the transfers from the point of interruption. For example, GridFTP allows partial

transfers to be resumed. 4) The module responsible for monitoring and adjusting concurrency collects dynamic transfer performance, and if significant discrepancies from the estimated performance are noticed, it adjusts the number of concurrency and parallel streams.

The Recovery phase interacts dynamically with the components of the execution phase to validate the completed request by collecting statistics, generating dynamic progress estimation on-demand, and validating transferred files at the end of the request. It has three functions. 1) It collects statistics from the execution of the replication request. 2) It generates dynamic progress estimation on-demand when a client asks for request progress status. This module needs the information on file transfers that completed, are in-progress, or are pending, as well as bandwidth usage statistics and estimation. 3) The file validation module can be running as soon as files are transferred, or at the end of the request, depending on the site preference. The reason for preferring file validation by checksum comparison after all transfers complete is that calculating checksums is computationally intensive and may perturb the running transfers. This module is also responsible for re-submitting files whose checksums indicated data corruption.

3.2 Transfer Queue Management and Balanced Concurrency

The BDM achieves high performance using a variety of techniques, including multi-threaded concurrent transfer connection management, transfer queue management and single control channel management for multiple data transfers, while the GridFTP library supports data channel caching and pipelining.

Transfer queue management and concurrency management contribute to more transfer throughput, including both network and storage. When there are many small files in the dataset, continuous data flow from the storage into the network can be achieved by pre-fetching data from storage on to the transfer queue of each concurrent transfer connection. This overlapping of storage I/O with the network I/O helps improve the performance.

Figure 2 shows the file size distribution from a dataset in IPCC CMIP-3 with 300GB of total dataset size. In this figure, most of the data files have less than 200MB of file size, and among those smaller files, 10-20MB file size range has the biggest portion. So, in most climate datasets, about one third of a dataset has less than 20MB in file size. As in Figure 3, BDM manages a DB queue for balanced access to DB from the concurrent transfer connections, and also manages the transfer queues for concurrent file transfers. Each transfer queue checks a configurable threshold for the queued total files size and gets more files to transfer from the DB queue

when the queued total files size goes below the configured threshold. Default threshold is set to 200MB based on the file size distribution as in Figure 2. Figure 4 shows the results from the effect of transfer queue and concurrency management. When the transfer queue and concurrency are well managed (shown in the bottom left plot of the figure), the number of concurrent data transfers shows consistent over time, compared to the ill or non-managed data transfers (shown in the top left plot of the figure), and it contributes to the higher overall throughput performance (shown in the bottom right of the figure).

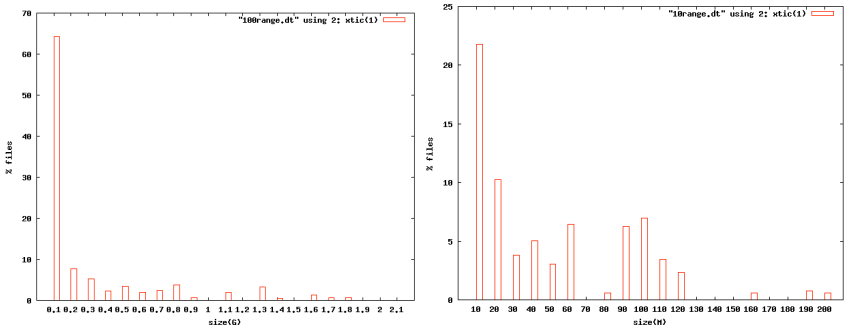

Figure 2. Typical file size distribution of a climate dataset in IPCC CMIP-3, showing majority of dataset file sizes less than 200MB (shown left), and majority of smaller files are around 10-20MB range (shown right)

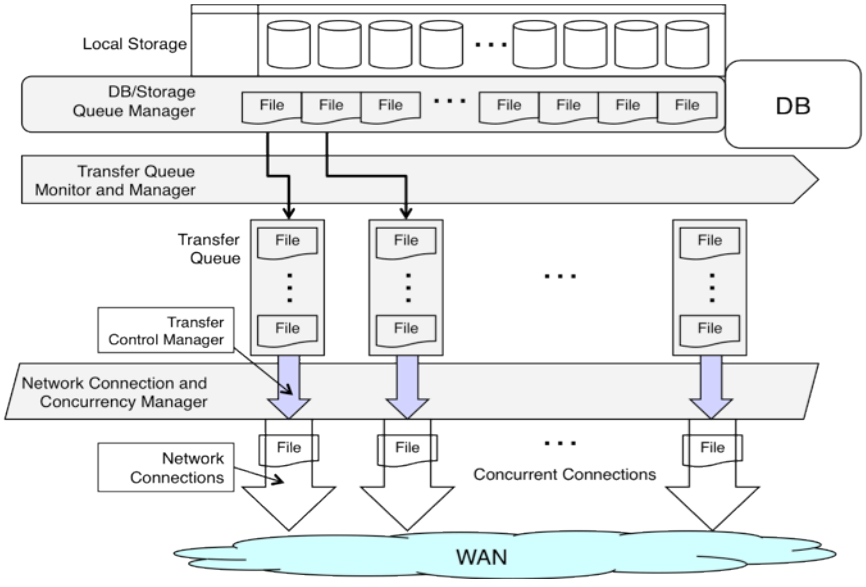

Figure 3. Transfer Queue Management and Concurrency in the BDM

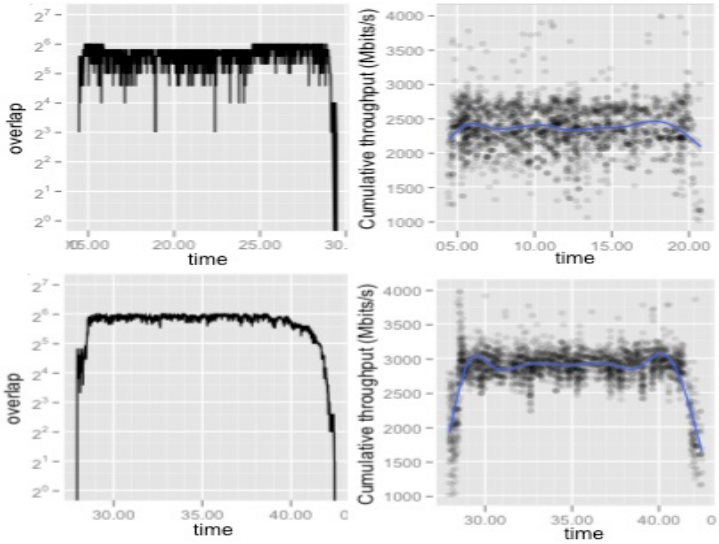

Figure 4. Results from transfer queue and concurrency management in BDM, showing ill or non-managed queue and concurrency on the top row and optimized queue and concurrency on the bottom row, for the number of overlapping concurrent transfers on the left and the transfer throughput over time on the right.

For further optimization in the transfer queue management, the order of the transfer files based on file sizes is considered in concurrent transfer connections. Also, the concurrent transfers are balanced across multiple source transfer servers, when multiple transfer servers are available at the data source. In the future, network delay will be considered in the calculation of concurrency and parallelism in BDM file transfers.

4. NetLogger and monitoring

The NetLogger Toolkit [4] implements a comprehensive end-to-end monitoring framework for performance analysis and troubleshooting of distributed applications. It includes the following components:

- *NetLogger methodology*: A methodology, summarized in the "Logging Best Practices" document [5], for analyzing distributed systems, which includes a simple, common message format for all monitoring events that includes high-precision timestamps.
- *NetLogger storage and retrieval tools*: Tools for distributed log collection, normalization, and storage in a relational database.
- *NetLogger visualization and analysis tools*: Python, SQL and R

programs to flexibly retrieve, visualize, and analyze NetLogger logs.

- *NetLogger client API library*: C/C++, Java, Python, Perl language bindings for source code instrumentation.

A unique feature of the NetLogger toolkit's client library is the "*log summarization"* module, which can perform adaptive summarization of high-volume events. In previous joint work with Globus, this module has been incorporated into GridFTP software to provide an online non-intrusive "*bottleneck detection"* capability for determining whether the transfer bottleneck is, at any given point during the transfer, the disk or network. The NetLogger storage and retrieval tools can ingest and correlate this information with logs from grid middleware and applications.

The goal of the monitoring is to track and help debug end-to-end performance. The tools that we used are NetLogger and Syslog-NG [6]. The monitoring is based upon GridFTP logs collected from the transfer servers. These logs are forwarded, by Syslog-NG, to a designated host, where NetLogger loads them into a database. A web interface provides a menu of plots that displays the combined information in the database. The end-to-end latency for the monitoring data, i.e. the time from the logging of the transfer by GridFTP to the visualization of the transfer performance in an online plot, is on the order of a few seconds. NetLogger analyses on file transfers provided valuable information on time-varying patterns in the overlap between multiple concurrent transfers for tuning the BDM queue and concurrency management algorithms.

5. Testbed

The Green Data Oasis (GDO) [7] at LLNL has over 600 TB of spinning disk and serves 45 TB of CMIP-3 multimodel data. Three GridFTP server nodes with Solaris 10 running ZFS on AMD-64 hardware were used with access to the 10 Gbps ESnet network. Two NERSC Data Transfer Nodes [8] were used to transfer data located on NERSC storage units based on GPFS. A 10 Gbps SDN through OSCARS could be reserved through ESnet between NERSC and LLNL.

In this test setup, randomly selected a few climate datasets from CMIP-3 were replicated for test runs under different transfer conditions. Dataset sizes range from 40 GB to 5 TB.

6. Results

Figure 5 shows a sample NetLogger graph from a series of short test runs run between NERSC and LLNL, at different times of day. The X-axis is time in seconds since the start of each transfer, and the Y-axis

shows MB/s of all currently overlapped transfers, which we call "cumulative MB/s". The cumulative MB/s is not a raw value: it is derived from the GridFTP "transfer" logs, which contain the time, duration and size of each transfer, by modeling each transfer as proceeding at the average bandwidth from start to finish. Every time a transfer starts, its average bandwidth is added, and every time one ends, it is subtracted. The resulting step function is actually a more accurate estimate of the transfer's throughput (goodput) than the router packet counters because it does not include retransmitted packets.

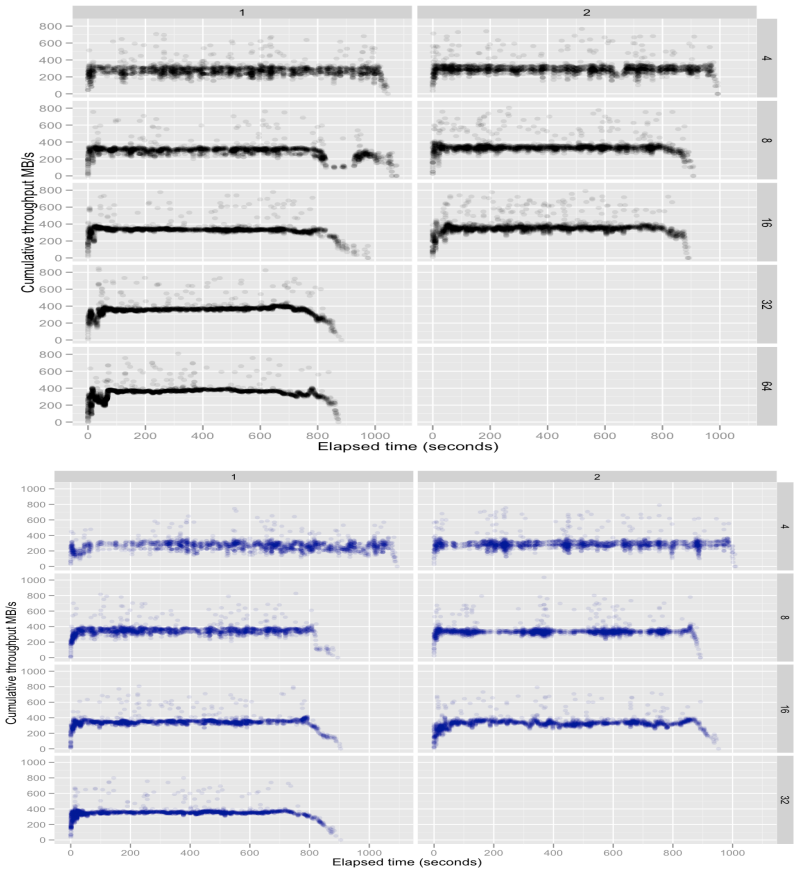

Figure 5. Test results showing cumulative throughput in MB/sec for different concurrency level and parallelism. Multiple tests were done at different time of the day and days.

For this path, the average bandwidth tended to improve at over 16 concurrent transfers and was consistently near the best at 32 streams. However, more tests need to be run across different network paths, storage systems, TCP stacks, etc., before we can create reliable heuristics for predicting optimal levels of concurrency and parallelism.

7. Summary

The climate community faces the difficult challenges of managing the distribution of massive datasets and accessing and analyzing them. The IPCC Coupled Model Intercomparison Project, phase 3 (CMIP-3) holds over 35 terabytes (TB) of data at the LLNL site. The IPCC Coupled Model Intercomparison Project, phase 5 (CMIP-5) is assembling together the most comprehensive archive yet today, and it is projected to be 10 petabytes (PB). These multi-model data sets and corresponding observation data sets will be distributed at many sites spanning six continents. The Earth System Grid (ESG) will be able to scale in order to handle increasing needs for data access in this highly collaborative decentralized environment, in relation to distributed data sharing processes associated with data discovery, access, movement, analysis, visualization and other computational resources. Bulk Data Mover (BDM) is to provide the efficient data delivery required for this scalability.

8. Acknowledgments

This work was funded in part by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under contracts DE-AC02-05CH11231.

9. References

[1] Earth System Grid (ESG), http://www.earthsystemgrid.org/
[2] Bulk Data Mover (BDM), http://sdm.lbl.gov/bdm/
[3] Williams et al., “The Earth System Grid: Enabling Access to Multimodel Climate Simulation Data”, in the Bulletin of the American Meteorological Society, February 2009.
[4] NetLogger, http://acs.lbl.gov/NetLoggerWiki/index.php/NetLogger_Toolkit
[5] Logging Best Practices, http://www.cedps.net/index.php/LoggingBestPractices
[6] Syslog-NG, http://www.balabit.com/network-security/syslog-ng/opensource-logging-system
[7] Green Data Oasis (GDO), https://computing.llnl.gov/resources/gdo/
[8] Data Transfer Node (DTN), http://www.nersc.gov/nusers/systems/datatran/